Chapter 13

Design of Multi-Agent Systems

In the context of LLM-based multi-agent systems (LLM-MAS), *collaboration goals* and *collaboration norms* serve as foundational elements that shape system behavior, interaction patterns, and overall effectiveness. Collaboration goals specify the explicit objectives agents aim to achieve – whether individually, collectively, or competitively – while collaboration norms define the rules, constraints, and conventions that govern agent interactions within the system. Together, these components establish a robust framework guiding effective communication, coordination, and cooperation among agents.

This section categorizes LLM-MAS into three broad classes based on distinct combinations of collaboration goals and norms: *strategic learning*, *modeling and simulation*, and *collaborative task solving*. Although not exhaustive, these categories cover a wide spectrum of LLM-MAS designs and clearly reflect how system objectives shape agent interactions and outcomes.

- **Strategic Learning** systems embed agents within a game-theoretic context, where agents pursue individual or partially conflicting goals. The interactions can be cooperative, competitive, or mixed, guided explicitly by predefined game rules and interaction norms. This setting often aligns with non-cooperative (strategic) and cooperative concepts in traditional game theory. Please refer to Section 13.1 for details.
- **Modeling and Simulation** contexts focus on agents acting independently, driven by diverse environmental or social factors. Here, interactions emerge organically without necessarily converging on common goals, reflecting the complex dynamics seen in large-scale social or economic simulations. Please refer to Section 13.2 for details.
- **Collaborative Task Solving** emphasizes systematic cooperation among agents to achieve explicitly shared objectives. Agents typically adopt structured workflows, clear role definitions, and highly predefined collaboration norms to synchronize their actions toward collective goals. Please refer to Section 13.3 for details.

In the remainder of this chapter, we elaborate on each category, examining how LLMs enable, influence, and enhance agent behaviors, interactions, and collective intelligence within our scope.

In the following, we examine these categories in detail, highlighting how each leverages the capabilities of large language models to shape agent behaviors and interactions.

13.1 Strategic Learning: Cooperation *vs.* Competition

Strategic learning refers to agents' capabilities to dynamically anticipate, interpret, and influence the actions of other agents within game-theoretic settings—whether competitive, cooperative, or mixed [949]. Agents iteratively adjust their strategies based on new information, commonly modeled using foundational concepts such as Nash equilibria [950], Bayesian games [951, 914, 952], or repeated interactions [953, 954]. With LLMs enabling nuanced linguistic reasoning, strategic learning increasingly integrates "soft" signals – including dialogue, persuasion, and implicit negotiation – thus enriching traditional game-theoretic reasoning frameworks [952, 955, 956, 957].

In economic applications, multi-agent strategic simulations provide valuable insights into market behaviors and negotiation tactics, highlighting both competitive and cooperative dynamics. For example, [958] and [951] demonstrate how LLM-empowered agents can simulate hiring processes, exhibit rational decision-making in controlled economic

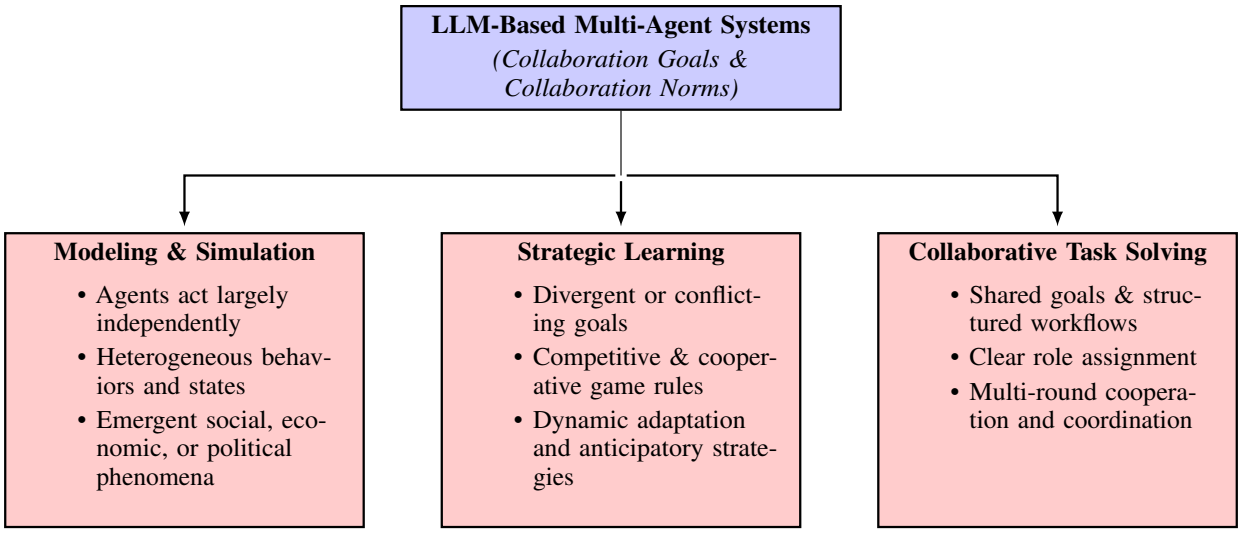

Figure 13.1: An overview of three major collaboration types in LLM-based MAS: *Modeling & Simulation*, *Strategic Learning*, and *Collaborative Task Solving*. Each category is distinguished by how agents' goals and norms are set (independent vs. divergent vs. shared) and how they coordinate.

experiments, and even forecast stock movements. [959] introduces a GPT-4-based competitive environment to illustrate how restaurant and customer agents compete to optimize profits and satisfaction, showcasing realistic bidding and pricing strategies. Meanwhile, [960] investigate Buyer–Seller bargaining in LLM-based negotiations, while [961] use ultimatum game simulations to illuminate policymaking decisions grounded in human-like strategic behavior.

Beyond conventional markets, strategic learning applies broadly wherever resource allocation, alliances, or competitive-cooperative trade-offs are present. Examples include multi-commodity competitions [962, 959], in which agents strategically negotiate terms to maximize individual benefits, or sustainability-focused contexts where agents coordinate resource consumption [963]. In gaming, social deduction games such as Werewolf, Chameleon, Avalon, and Jubensha require agents to manage the complex interplay between deception and collaboration [964, 965, 966, 153, 919, 967, 968, 969, 970]. Studies by [971, 965] highlight LLM-based agents that excel at orchestrating subtle deceit and collaboration, while [967, 972, 968, 969] emphasize adaptive, multi-round strategy in Avalon. [970] further pushes this boundary by showcasing autonomous, multi-agent interactions in the *Jubensha* murder mystery genre, re-creating complex narratives. Similarly, diplomatic simulations ([973] and [974]) employ LLM-based agents to emulate sophisticated geopolitical negotiation and alliance formation dynamics at global scales.

Summary A key advantage of LLM-driven strategic learning lies in effectively combining rigorous game-theoretic logic with natural language reasoning. This fusion enables agents to interpret sophisticated instructions, engage in persuasive dialogue, and adapt more flexibly to novel or unstructured settings. Consequently, LLM-based strategic agents hold significant promise for accurately modeling complex real-world interactions – spanning economic competition, social negotiation, and geopolitical strategy – far more effectively than conventional rule-based or numeric-only approaches.

13.2 Modeling Real-World Dynamics

Modeling and simulation represents another crucial area of application for LLM-based multi-agent systems (LLM-MAS), aiming to replicate complex social, economic, and political phenomena at scale. By utilizing LLMs' sophisticated language understanding and contextual reasoning, these simulations can feature highly heterogeneous agents whose evolving behaviors mirror real-world dynamism. Unlike strategic learning environments that emphasize explicit competitive or cooperative goals, agents in modeling and simulation scenarios operate independently, guided by their domain-specific roles, preferences, and interactions with the simulated environment [975].

In healthcare, for example, [921] introduces *Agent Hospital*, where LLM-powered doctor agents iteratively refine treatment strategies through realistic interactions with virtual patients. This enables researchers to test management protocols, training paradigms, and "what-if" scenarios in a controlled yet realistic setting. Similarly, in economic contexts, [976] present *EconAgents*, leveraging LLM-driven agents to realistically model individual-level behaviors such as employment decisions, consumption patterns, and savings strategies. These agents facilitate expressive macroe-

conomic simulations, surpassing traditional numeric or strictly rule-based methods in adaptability and realism [977]. In addition, political science applications also benefit from this approach. For example, [978] and [977] successfully simulate election processes and policymaking dynamics, revealing how public discourse, candidate strategies, and voter interactions shape real-world political outcomes.

Beyond economics and politics, LLM-based simulation accommodates a variety of social and cultural phenomena. For example, [979] and [255] use simulations of linguistic and emotional propagation in social networks to investigate how opinions, beliefs, or sentiment clusters form online. Research by [980] explores how opinion dynamics evolve under various topological and interaction patterns, while [981] examines the conditions under which fake news spreads or stalls in heterogeneous agent populations. Large-scale simulation platforms such as GenSim [982] and OASIS [936] push the boundary further by scaling to tens of thousands or even millions of user agents, thus enabling the study of emergent group behaviors and systemic effects—such as viral information diffusion, echo-chamber formation, or group polarization—under realistic constraints.

Summary The strength of LLM-based simulation lies in capturing both the structural dynamics (e.g., network topology or institutional rules) and the cognitive or linguistic nuances that drive real-world behavior. By embedding language-based reasoning into agent models, researchers can examine complex social processes—like persuasion, framing, or cultural transmission—that would be difficult to capture through purely numeric or rule-based approaches.

13.3 Collaborative Task Solving with Workflow Generation

Collaborative task solving orchestrates multiple agents toward a clearly defined objective through structured workflows. In contrast to strategic learning (which may involve competing interests) or open-ended modeling and simulation (where agents act independently), collaborative agents function as part of a unified problem-solving pipeline. Agents typically follow clearly defined roles (e.g., "Planner", "Implementer", or "Evaluator") and stage-based processes to ensure efficient and accurate task completion.

Systems such as MetaGPT [626], CAMEL [848], Communicative Agents [983], and frameworks described in [924] exemplify how clearly defined roles, responsibilities, and decision flows allow LLM-based agents to coordinate effectively. A typical workflow might involve one agent analyzing a problem statement, another proposing a solution outline, a third implementing partial solutions, and a fourth verifying correctness. Communication among these agents is often carried out through iterative rounds of natural language "dialogue", leveraging the inherent language-generation strengths of LLMs. This structured approach also proves beneficial for scaling to more ambitious projects, as sub-tasks can be delegated to specialized agents with domain-specific prompts or training.

Recently, collaborative task-solving systems have been explored extensively in software development scenarios (e.g., multi-agent coding, debugging, and testing). However, scientific discovery represents a particularly prominent and compelling application. For example, the *Agent Laboratory* [746] employs agents in structured scientific workflows: proposing hypotheses, designing experiments, analyzing results, and refining subsequent inquiries, which effectively mirrors the iterative nature of the scientific investigation. Similar multi-agent designs can be adapted to tasks such as literature review, policy drafting, or large-scale data analysis, using well-defined protocols to maintain coherence and avoid duplication of effort.

Summary Compared to other LLM-based multi-agent paradigms, collaborative task-solving inherently prioritizes clarity and predictability: Each agent's role and objective are predefined, limiting emergent or chaotic behaviors. This structure is particularly advantageous in domains requiring precision, accountability, or sequential decision-making. At the same time, research is ongoing to strike the right balance between structure and flexibility, which ensures that agents have enough autonomy to creatively contribute solutions while adhering to a shared workflow that ultimately guarantees reliable, high-quality task completion.

Discussion The aforementioned three dimensions—*strategic learning*, *modeling and simulation*, and *collaborative task solving*—reflect the breadth of LLM-based multi-agent systems. Each category addresses distinct research questions and real-world applications, leveraging language-based reasoning to tackle challenges that extend beyond the capabilities of conventional, purely numeric, or rule-driven agent designs.

13.4 Composing AI Agent Teams

In MAS, agents are the core units that interact within the system and are critical to its functionality. These agents can be categorized as either homogeneous or heterogeneous, depending on whether they share identical or differing personas, capabilities, and action spaces.

Homogeneous Homogeneous agents that share identical capabilities, action spaces, and observation spaces. Compared to single-agent systems, the primary advantage lies in task parallelization, allowing multiple agents to handle different parts of a task simultaneously and improve overall efficiency. They are often used in simpler, coordinated tasks where uniformity across agents can drive improved performance.

Several studies have applied homogeneous agents to simulate teamwork in games like Overcooked and Minecraft, as well as real-world tasks such as household labor division. [924] proposed a cognitive-inspired modular framework that enables LLM-based agents to communicate through natural language to perform labor division, request assistance from one another, and collaboratively complete object transportation tasks. [984] introduced prompt-based organizational structures into the framework, reducing communication costs between agents and improving team efficiency in household tasks such as preparing afternoon tea, washing dishes, and preparing a meal. Furthermore, several studies [926, 925] have employed multiple LLM-based agents in popular games such as Overcooked and Minecraft to experiment with their ability to cooperate and complete tasks. According to the game settings, these agents are also homogeneous.

Heterogeneous Agent diversity plays a crucial role in improving collaboration outcomes. Research shows that heterogeneity among agents can enhance problem-solving capabilities, as diverse agents bring varied perspectives and skills to the task at hand [985, 986]. Heterogeneity contributes to richer problem-solving strategies and improves overall collaboration in MAS. The heterogeneous characteristics of agents can be reflected in the following dimensions: personas-level heterogeneity, observation-space heterogeneity, and action-space heterogeneity. Note that these heterogeneities are not mutually exclusive—a heterogeneous agent may exhibit one or more of these characteristics.

- *Personas-level heterogeneity.* Refers to diversity in agent profiles, which influences how agents approach problem-solving and interact with one another. Most current LLM-based heterogeneous multi-agent systems fall into this category [987, 627, 50, 970]. For example, in software development, agents may take on personas such as programmers, product managers, or testers. In medical diagnostics, agents may represent cardiologists, oncologists, or paediatricians, each with distinct areas of expertise. The distinct perspectives and expertise of each persona contribute to more robust decision-making. While these heterogeneous agents may share the same action space—such as writing documents [626] (e.g., code, requirement reports, or test reports) or providing diagnostic advice [922]—their personas influence the outcomes of these actions, where role-specific enhancements within multi-agent architectures have shown to significantly streamline and optimize task execution. For instance, a product manager performing the action of writing a document would produce a requirements report, whereas a programmer performing the same action would produce software implementation code [626]. This diversity leads to better decision-making and innovation, especially in complex, multidisciplinary tasks.
- *Observation-space heterogeneity.* In MAS, the ability of agents to perceive and interpret their environment can vary. Observation-space heterogeneity refers to these differences in what agents can observe or perceive within their environment. For example, in the game Werewolf, some agents, like werewolves, can see the identities of their teammates, and the seer can obtain the identity of a designated player, while others, like villagers, cannot see the true identity of any player [971]. Similarly, in the Avalon game, different roles have distinct observation spaces [919, 972], thus influencing the strategies and communications of the players. In these settings, each agent's perceptual ability or observation space is directly linked to their role in the system. In a multi-agent system, this variation in what agents can observe often influences their decision-making, communication, and coordination with other agents.
- *Action-space heterogeneity.* On the other hand, this refers to fundamental differences in the actions agents can perform due to physical or functional constraints. This is particularly relevant in both virtual and physical environments where agents may have different capabilities based on their design or purpose. In the virtual environments of games like Werewolf [965, 971, 966] and Avalon [919, 967], different roles have distinct abilities or skills [971, 919, 972]. For example, in Werewolf, while werewolves may have the ability to communicate secretly with each other, villagers might be limited to voting or observing only. This dynamic requires agents to collaborate based on their unique capabilities and promotes the learning of strategies such as teamwork, trust, and deception in their interactions. Meanwhile, in robotics, agents may exhibit diverse physical capabilities. For instance, as described in [988], some robots lack mobility and can only manipulate objects, while others are specialized for movement but cannot manipulate objects. In such cases, agents with different action spaces must divide tasks effectively, leveraging their specific abilities to take on the parts of the task they are suited for, ultimately collaborating to complete the overall task. This type of heterogeneity requires agents to collaborate and coordinate their actions efficiently, often dividing tasks based on their individual strengths.

Homogeneity to Heterogeneous Evolution In some LLM-based multi-agent systems, agents have the ability to evolve autonomously and continuously adapt through interactions with their environment. Due to the inherent randomness

in both LLM models and the environment, the evolution of these agents often follows different trajectories. This can lead to heterogeneous behaviors emerging over multiple simulations, even when agents initially have homogeneous personas and action spaces. For example, as shown in [989], agents with identical action spaces and personas at the start developed differentiated roles after multiple rounds of interactions with the environment and other agents. Some agents, for instance, specialized in food gathering, while others focused on crafting weapons. Similarly, [990] observed that initially homogeneous agents developed distinct language usage patterns, emotional expressions, and personalities after group interactions. These emergent behaviors demonstrate the possibility of transitions from homogeneous to heterogeneous systems.

13.5 Agent Interaction Protocols

In this section, there will initially be classification of typical kinds of messages, providing a clear view regarding the content and exchange modes for agent interactions. Next, agent-environment, agent-agent, and agent-human communications interface designs will be addressed. Architectural issues and protocol specifications for transparent information exchange will also be addressed. Interface standardization will have a special focus, which is essential for providing interoperability, scalability, and efficiency for multi-agent systems. The section will end with unification of communication protocol discussions, where agent-environment or agent-user interacting design principles and requirements are addressed, as well as providing clarity, consistency, and functional coherence for various applications for LLM-based systems.

13.5.1 Message Types

Structured: Structured messages, either in JSON ([991, 992]), XML ([993, 636]), or as a code ([626, 627, 994]), are a crucial aspect of multi-agent system communication with LLM. The primary advantages of structured messages are their syntactically and semantically defined structure, enabling unambiguous understanding and straightforward parsing. With their lack of ambiguity, they facilitate unerrant information extraction and processing with much less overhead on computation and greater system dependability. For example, JSON and XML can represent specific-task configuration parameters or facilitate data exchange as a machine-readable mode, and messages written as a code can even be executable several times directly, which makes workflow and automation simpler.

Structured messages are particularly well-suited for high-efficiency, deterministic applications. They are useful for sub-task decomposition, sub-task assignment, and coordination among agents for cooperative multi-agent architecture because they explicitly state operational commands. Moreover, as structured messages have a prescribed form, retrieving data as well as storing data is facilitated and system optimization and longitudinal analysis are also feasible.

Unstructured: In contrast, unstructured messages, e.g., natural text ([971, 970, 919]), visual data, e.g., images, videos, and audio signals, e.g., speech, ambient sounds ([995, 996, 762]), have higher information density and representational capability. Such modalities are best suited for communication with nuanced and context-dependent information. Images, for instance, communicate spatial relationships, illumination, and facial expressions, and videos communicate dynamic temporally-organized sequences, e.g., state or behavior changes over time. Similarly, audio signals also communicate not just linguistic information but also paralinguistic information, e.g., tone, emotion, and intonation, which are critical for natural and context-aware interactions.

Unstructured messages are well-adapted for ambiguity tasks, as well as for complex, real-world settings. The fact that they can express abstract ideas as well as affective subtlety, or implicit contextual suggestions, makes unstructured messages well-suited for creative, as well as discovery-oriented, problem spaces. Unstructured data's complexity, however, calls for advanced processing techniques, for example, feature extraction based on deep learning, for one to tap into their full potential. Advances with pre-trained LLMs as well as multi-modal large language models have alleviated these complexities to a large extent, enabling novel applications for unstructured communication within multi-agent systems [533, 513, 997].

Summary: Unstructured and structured messages have complementary roles for multi-agent communication with LLM-based. While structured messages offer accuracy, consistency, and computation efficiency and are appropriate for operational and deterministic operations, unstructured messages offer rich, contextualized representations enabling agents to negotiate vague, creative, highly dynamic situations. Together, these modes offer a foundation for adaptive, effective multi-agent cooperation.

13.5.2 Communication Interface

Agent-Environment Interface LLM-based agents will typically have to act on their environment once or several times in order to perform a range of operations. From the agent's point of view, its output into the environment is something that it would prefer, e.g., a UI click, web request, or a move for a computer graphic's character. Environments differ with regard to what actions they will accept, and so as not have its actions not get executed, the agent must find out what actions are for a specific environment that it is acting within and perform actions that are for a specific task as well as valid for a specific environment. After the agent outputs its chosen action, the agent will have a return from the environment. It will consist of observations if successful, or a feedback on error if there was one. The agent will have to act on this feedback. There are nowadays various types of environments where an agent can act, e.g., operating systems, computer games, database, and e-commerce websites. To make agent-environment interfaces share a common interface and have agents trained on various LLMs plug into various environments with minimal further adaption, various frameworks have been proposed. These frameworks make for easier tests on agents' capability on various executable environments [706].

Agent-Agent Communication In MAS, communication through natural language is predominant. This is likely because large language models possess strong linguistic capabilities due to pretraining on massive natural language corpora. Another possible reason is that, for many tasks, natural language communication is already sufficient to meet the requirements. Based on the type of information exchanged, multi-agent systems can be categorized as follows: *Natural Language-Based Systems* Among LLM-based multi-agent systems utilizing natural language, text-based communication is the most common [922, 924, 987, 970, 998]. There are also some systems that use voice as the medium of communication [996, 762, 999, 1000]. In these systems, agents engage in behaviors such as discussions, negotiations, persuasion, or critique through natural language to achieve their objectives. *Structured Information-Based Systems* Compared to natural language, structured information has characteristics such as higher consistency, lower parsing complexity, and reduced ambiguity, making it more suitable for efficient and low-cost communication between agents [626]. In some implementations, the information exchanged between agents is structured into distinct components to facilitate easier parsing and utilization by the receiving agent. For instance, the exchanged information might include fields specifying the `sender`, `receiver`, `message type`, and instructions on how the recipient should parse or use the `content` [929].

Human-Agent Communication The purpose of developing multi-agent systems is to expand the boundaries of human capabilities and cognition, ultimately serving human well-being. While in some social simulation multi-agent systems, humans primarily exist as observers [50, 1001], most multi-agent systems allow human participation in various forms. During this participation, humans need to communicate with agents, and this communication can take the form of either natural language or structured information [924, 930]. When human-to-agent communication primarily relies on natural language, a single LLM often acts as a hub to parse human natural language into structured information that agents can process more effectively for subsequent operations. This hub LLM can either exist within the multi-agent system or function independently of it. To save time and enhance communication efficiency, humans can also use structured information to communicate with the multi-agent system through programming or similar methods. By following predefined communication protocols, humans can send messages containing the required data to the multi-agent system. The system will then process the messages and data according to its internal logic and return the results. [931]

13.5.3 Next-Generation Communication Protocols

The field of LLM-based agents is still in its infancy. Developers typically design agent architectures and communication mechanisms tailored to specific domains or tasks, including agent-to-environment, agent-to-human, and inter-agent interactions. However, most existing systems lack a unified communication framework, resulting in fragmented, siloed ecosystems. Multi-agent systems, tools, environments, and data sources often operate independently, making it difficult for agents to interoperate or share capabilities. Furthermore, the burden of learning and implementing bespoke protocols falls on humans, and almost all current protocols are manually designed—a labor-intensive process that often lacks semantic flexibility or scalability.

To address these issues, several new agent communication protocols have been proposed, each targeting different aspects of the protocol design stack.

Internet of Agents (IoA) [933] introduces an internet-inspired, instant-messaging-like communication architecture that supports dynamic team formation and task-driven collaboration. Agents register with a central coordination server, which handles identity management and discovery. Communication flows are orchestrated using FSM (Finite State Machine)-based dialogue templates. IoA supports multiple message types, including discussion, task assignment, and triggering mechanisms, and provides structured fields for controlling speaker turns, nested group formation, and

maximum dialogue length. This allows agents to select and adapt message formats to match specific coordination phases, offering flexibility within a fixed schema.

Model Context Protocol (MCP) [931], developed by Anthropic, focuses on enabling LLM agents to access structured tools and data. It adopts a fully centralized approach based on OAuth identity authentication, and interactions are constrained to JSON-RPC 2.0 messages. While it lacks a meta-protocol layer or semantic negotiation capabilities, its simple and rigid architecture makes it a practical choice for tool use cases with well-defined APIs. However, MCP sacrifices flexibility and extensibility, requiring manual registration of supported functions.

Agent Network Protocol (ANP) [1002] aims to achieve full decentralization. Agents identify themselves through W3C-compliant decentralized identifiers (DIDs) and communicate over encrypted peer-to-peer channels. The protocol includes a meta-protocol layer that enables agents to negotiate which application-level protocol to adopt, supporting semantic protocol selection based on agent capabilities. ANP also allows for multi-protocol support at the application layer (e.g., HTTP, JSON-RPC, natural language), providing strong extensibility and decentralization but does not yet explicitly support public protocol reuse.

Agora [932] offers a highly flexible and language-driven protocol mechanism. Instead of registering pre-defined APIs, agents can generate and share Protocol Descriptions (PDs), which are free-text descriptions of communication semantics. Using a large language model, agents can dynamically interpret and execute any PD at runtime. This allows protocols to be created, deployed, and used entirely through language, without any manual registration or configuration. Agora avoids centralized registries and supports decentralized protocol sharing: agents may publish or retrieve PDs from peer-distributed repositories to enable cumulative learning and interoperability across systems.

Summary: As shown in Table 13.1, next-generation agent communication protocols differ along key dimensions such as identity and security mechanisms, meta-protocol negotiation capabilities, application-layer flexibility, and the degree of centralization. A unified, secure, scalable, and dynamic protocol infrastructure—where agents can negotiate and co-create protocols on the fly—is critical for enabling large-scale, interoperable agent ecosystems. While current frameworks such as MCP, ANP, Agora, and IoA represent early but promising steps, protocol design remains a rapidly evolving frontier in the development of intelligent agent systems.

Table 13.1: Comparison of four agent communication protocols (MCP, ANP, Agora, IoA) across identity, negotiation, and execution layers.
PD = Protocol Description; **DID**:Decentralized Identifier; **LLM**:Large Language Model; **FSM**:Finite State Machine.

Layer	MCP	ANP	Agora	IoA
Identity & Security	OAuth-based centralized identity authentication.	DID-based decentralized identity with encrypted channels.	No centralized registration. Identity derived from PD hash.	Agents register with a central server for identity and discovery.
Meta-Protocol Layer	No meta-protocol layer; relies on pre-defined interfaces.	Uses DID document to negotiate and select appropriate protocol via semantics.	LLM interprets PD text to automatically negotiate and deploy communication protocols.	A centralized discovery mechanism combined with FSM-based dialogue flow control.
Application Protocol Layer	Supports only JSON-RPC 2.0.	Supports multiple protocols such as HTTP and natural language.	Allows arbitrary PD-driven protocols with high flexibility.	Task-driven protocol coordination supporting multiple message formats.
Degree of Centralization	Highly centralized architecture.	Fully decentralized.	Decentralized: no registration or fixed ID, with optional peer-to-peer PD sharing.	Highly centralized architecture with a central coordination server.
Protocol Flexibility	Fixed and rigid; hard to adapt beyond JSON-RPC.	Highly flexible with semantic negotiation.	Extremely flexible; any PD can define a new protocol dynamically.	Moderately high flexibility; agents can select and adapt message formats based on task phases and coordination needs.

Table 13.2: Classification framework for LLM-based multi-agent systems, highlighting different aspects of system design, communication, collaboration, and evolution. Below are our abbreviations, for ease of reference:
M&S = Modeling & Simulation, CTS = Collaborative Task Solving, SL = Strategic Learning, S-D = Static-Decentralized, S-L = Static-Layered, Hom = Homogeneous, Het = Heterogeneous, T/M = Teaching/Mentoring, C-O = Consensus-Oriented, T-O = Task-Oriented, CL = Collaborative Learning, Dict = Dictatorial, D-B = Debate-Based, CI = Collective Intelligence, Ind = Individual.

Paper	**System Design**	**Communication**			**Collaboration**		**Evolution**
	Category	**Typology**	**Interface**	**Agent Type**	**Interaction**	**Decision**	**Type**
Agent Hospital [921]	M&S	S-D	Text	Het	T/M, C-O	Dict	Ind
Welfare Diplomacy [934]	M&S	S-L	Code, JSON, Text	Hom	CL	Voting	CI
MEDCO[923]	M&S	S-L	Text	Het	T/M, C-O	Dict	Ind
MedAgents[922]	M&S	S-L	Text	Hom	T-O	Dict	CI
Generative Agents [50]	M&S	S-D	Visual	Hom	CL	Dict	Ind
RECONCILE [918]	SL	S-D	Text	Hom	CL	D-B	CI
Agent Laboratory [746]	CTS	S-L	Code, Text	Het	C-O, T-O	Dict	Ind
CoELA[924]	CTS	S-D	Text	Hom	T-O		
The virtual lab [752]	CTS	S-L	Text	Het	C-O, CL	Dict	Ind
SciAgents [743]	CTS	S-L	Text	Het	T-O	Dict	CI
S-Agents [927]	CTS	S-D	Text	Het	T-O, CL	Dict	
GPT-Bargaining [1003]	CTS	S-D	Text	Het	C-O	D-B	CI
FORD [1004]	M&S	S-D	Text	Het	C-O	D-B	CI
MADRA [1005]	CTS	S-D	Text	Het	C-O	D-B	
Multiagent Bench [948]	CTS	S-D	Text	Hom	T-O, CL	D-B	CI, Ind
OASIS [936]	M&S	D	Text	Het	C-O		
S^3 [255]	M&S	S-D	Text	Het	C-O		
FPS [981]	M&S	S-D	Text	Het	C-O		
GPTSwarm [1006]	CTS	D	Code, JSON, Text	Hom	T-O	Dict	CI, Ind
ChatEval [1007]	CTS	D	Text	Hom	T-O	Voting	CI
MetaGPT [626]	CTS	S-L	Code, JSON, Text, Visual	Het	T-O	Dict	CI
AutoAgents [1008]	CTS	D	Text	Het	T-O	C-O	CI
SWE-agent [628]	CTS	D	Text	Hom	T-O	Dict	Ind
AgentCoder [994]	CTS	D	Code, Text	Het	T-O	D-B	CI
MASTER [1009]	CTS	S-L	Text	Hom	T-O	D-B	CI
Reflexion [48]	CTS	D	Text	Het	T-O	D-B	Ind
MACM [1010]	CTS	D	Text, Code	Het	T-O	D-B	CI
Debate [985]	CTS	S-D	Text	Het	C-O	D-B	CI